Event Queue Compare

Diagnostic LA

Fri, Apr 15, 2005

While studying the clock decoding behavior of the Digital PMC board that is used in all the Linac PowerPC front ends, a diagnostic tool was written, called `EVQC`, for comparing the clock events seen by two nodes. This note describes its functionality. Its parameters are:

`ENABLE`	`B`	Usual enable Bit#
`NODE1`		Reference node#
`NODE2`		Target node#

The program logic monitors the `EVTLOG` data stream, to which an entry is written for every clock event received by the node, in both nodes. (This data stream is assumed to be at `DSTRM` table index 4.) The 15 Hz replies from each node are processed by accumulating counts of each event seen. Then the accumulated counts for each event are compared and the difference checked for change. When there is a change, a new log entry is made, including the event number, the change in the count difference, and the time of day to a resolution of 15 Hz cycle. One log entry occupies 8 bytes.

Since the replies from each node are not necessarily captured at exactly the same instant, one cannot be sure that all counts should match comparing the two replies received within a single cycle. To deal with this, an entry that is to be logged is placed into a temporary queue, for confirmation during the processing on the following cycle. If a change in the opposite direction is also seen on the next cycle, then the log entry is canceled; only queued log records are logged that have lasted through the next cycle.

The logic used to implement the above confirmation step uses a queue that has one `IN` pointer and two `OUT` pointers, called `OUT1` and `OUT2`. The `IN` pointer is used by the the writer. The `OUT1` pointer is used by one reader to advance toward `IN`, and the `OUT2` pointer is used by another reader to advance toward `OUT1`. This method is used many times in the system code for the front ends. It is logically equivalent to using two queues in series.

Another wrinkle can occur if nonzero status is returned from the `GetDat` call that retrieves the data each cycle. To that end, the time of the last nonzero status is recorded, along with a count of the number of times it occurs. This is a problem, because there is no way to "retry" a reply from a data stream, as each successive periodic reply effectively "moves along" through the data stream. Ignoring the replies on a cycle that gave nonzero status may or may not cause in invalid log record, depending on whether the records for that cycle, had they been received correctly, actually match. The cause of such nonzero status is not completely known, although it may imply something to do with the network itself.

The logged results typically show that on the same cycle, one extra event is seen and a different event is missing. This is consistent with the missing event being misread in the target node. Note that the logged results assume that the reference node is correct. If the reference node `EVTLOG` data stream is suspect, then it makes it harder to guess which node is in error. A typical pair of logged records is as follows:

```
1DFF 0417 1543 3608   missing 1D event April 17, 1543:36, cycle 08.
EB01 0417 1543 3608   extra EB event seen by target node
```

The interpretation of this pair of logged records is that a `1D` event was misread as a `EB` event. One can use the Print Memory PA to list the log records from the LA's static memory block.